FH·W-S

iOS Programmierung

(mit Swift)

Peter Braun, Florian Bachmann & Andreas Wittmann
@pe_braun @florianbachmann @anwittmann

Mobile Solutions - Deutsche Telekom AG
FHWS - Hochschule für angewandte Wissenschaften Würzburg-Schweinfurt

#FHWSSwift

Agenda

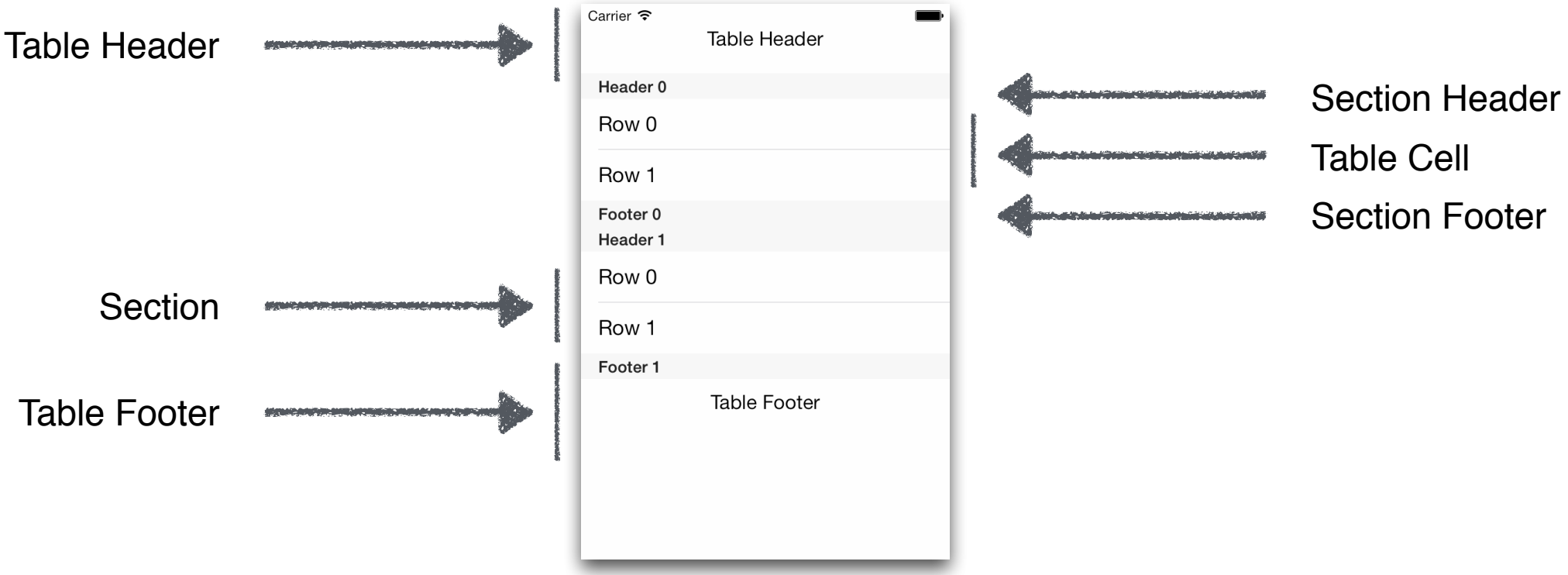
1. **Introduction** – Organisatorisches
2. **First iOS-Project** – Hello World, **First iOS-Project** – Still Hello World (now with Code 😄)
3. **Swift**, Wait!, What about Objective-C?, Why Swift?
4. **A (not so) Quick Tour**
5. **Documentation**
6. **The basics –** iOS Architecture & more
7. **User Interfaces –** View Controller, Auto Layout & Size Classes
8. **Storyboard & Segues**
9. **Tables & NavigationController**
10. **TabBarController**
11. **Notifications**
12. **PickerViews**
13. **Touches, Gestures, 3D Touch, Peek & Pop**
14. **ScrollView & StackViews**
15. **Networking** – JSON & Dependency Managers
16. **WebKit**
17. **Maps**
18. **Storage & Data persistency –** NSUserDefaults, NSKeyedArchiver & Core Data
19. **ObjC**

Tables
UITableViewController, UITableView & Cells

`UITableView`

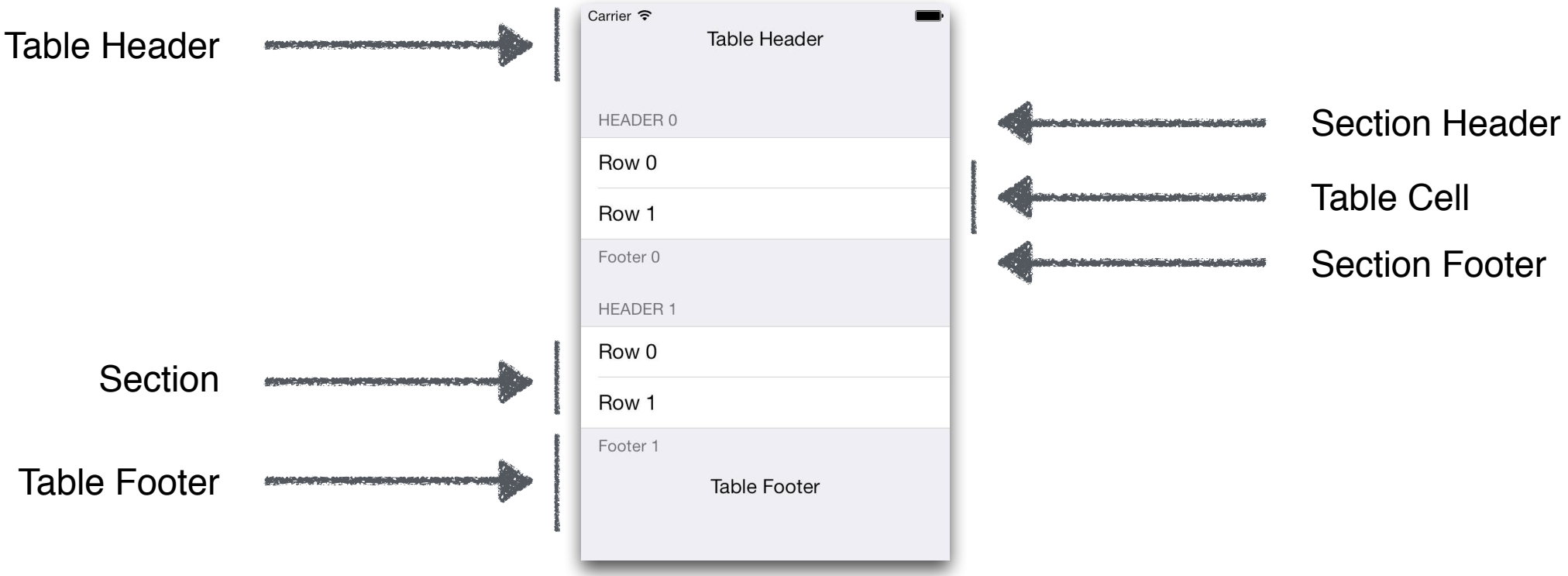
- Displays a scrollable list of items in a single column
  can handle a large amount of data performant
- Each column is a individual item of a `UITableViewCell` object
- `UITableViewCells` are used to draw the Columns
  complete customizing of cells possible
- Different optional styles of a table view are available
  zero or more sections, header and footer
- the cell can be ordered plain or grouped

UITableView - **Plain** Style



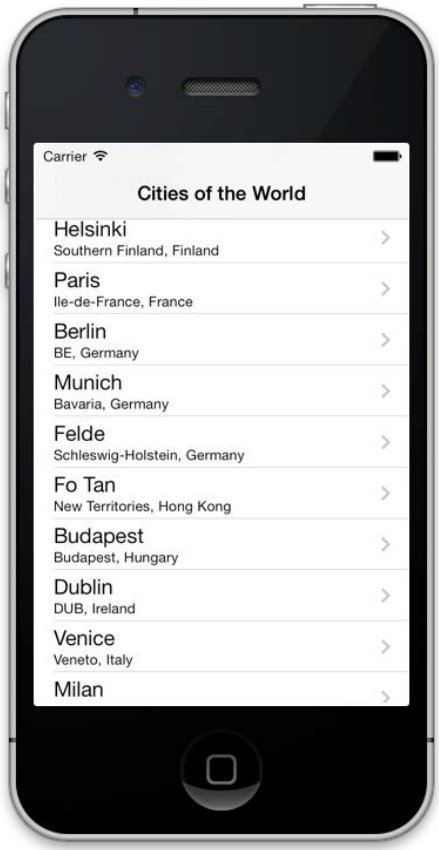
Source: Stanford CS193p

`UITableView` - **Grouped** Style

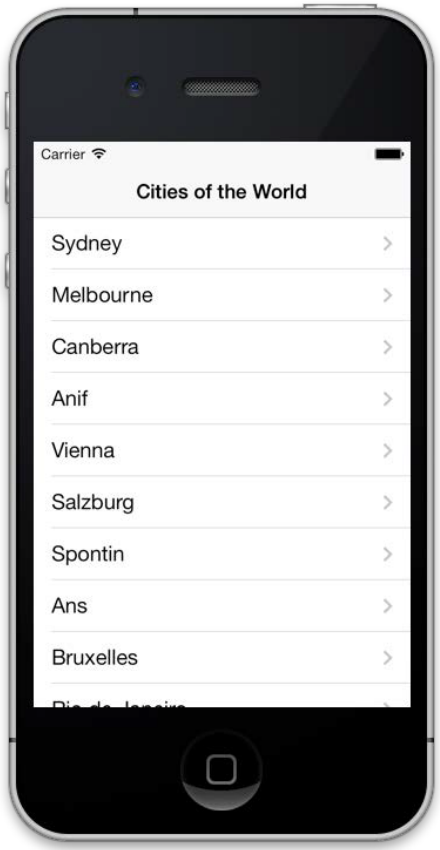


Source: Stanford CS193p

The 4 Default Cell-Types

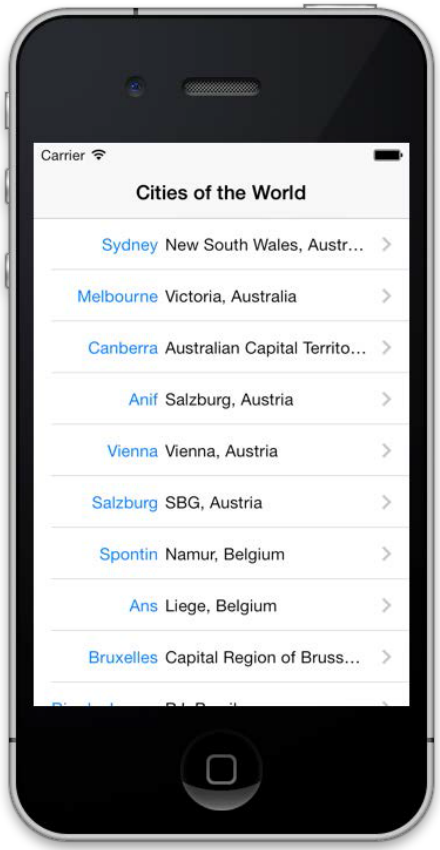


Subtitle



Basic

Right Detail



Left Detail

`UITableView`

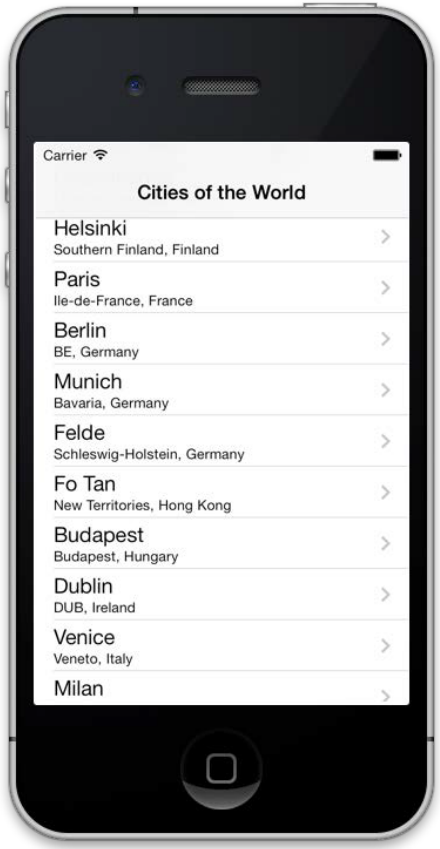
- *What is the `Delegate`?*
  Controls how the table is displayed
- *What is the `DataSource`?*
  The datasource provides what is displayed in the cells

Common Patterns - Delegate Reminder

- Delegation is a design pattern
- it enables a class or structure to delegate (hand off) some of its responsibilities to an instance of another type
- it is often referred as **loose coupling**
- many Apple Controls need to implement the provided delegate methods (`UITableView`, `UITextField`, `UIWebView`, `UIActionSheet`, `UIGestureRecognizer`, `UIScrollView`, ...)
- `DataSource` is basically a delegate!
  - `Delegate` controls the user interface
  - `DataSource` is a 'delegate' of control data
    - (as seen before `UITableView` can have both)

Source: Apple: Protocols

UITableView (can be a view in an UIViewController)



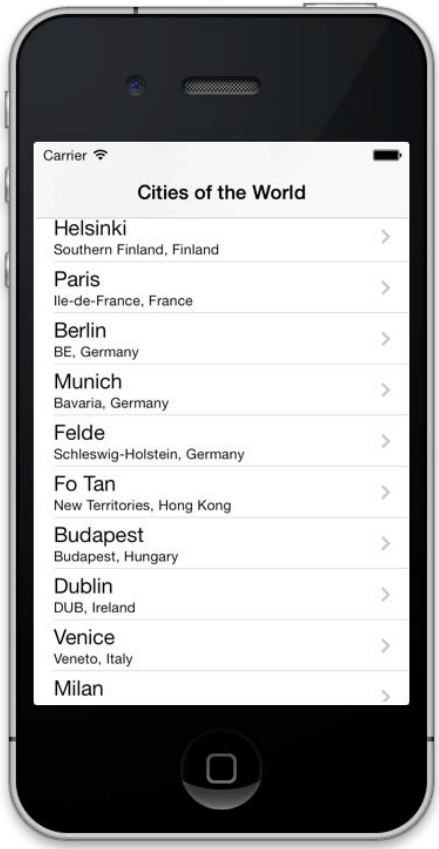
control data

Data Source

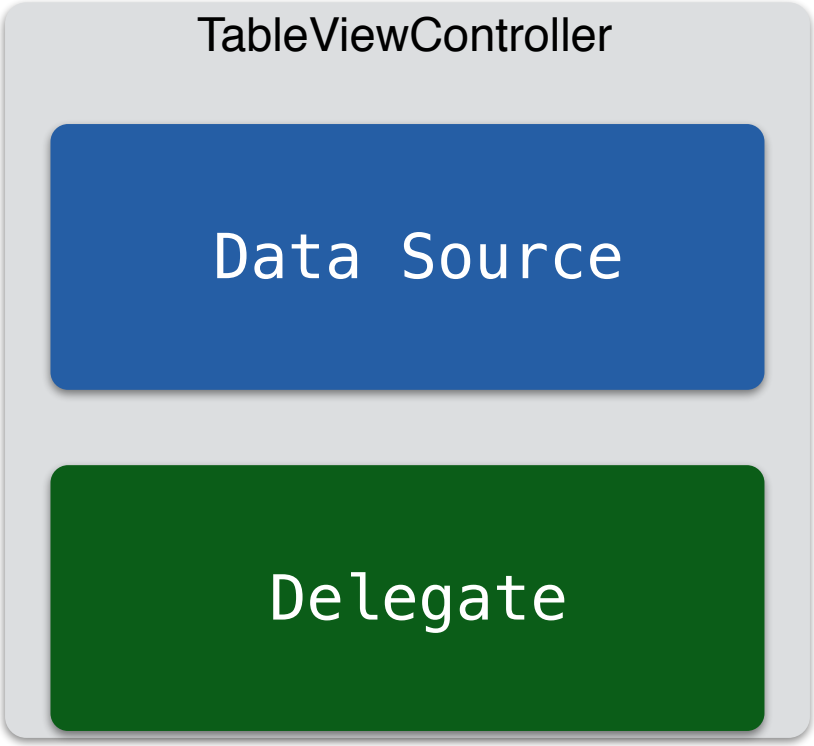
tap actions

Delegate

UITableViewController

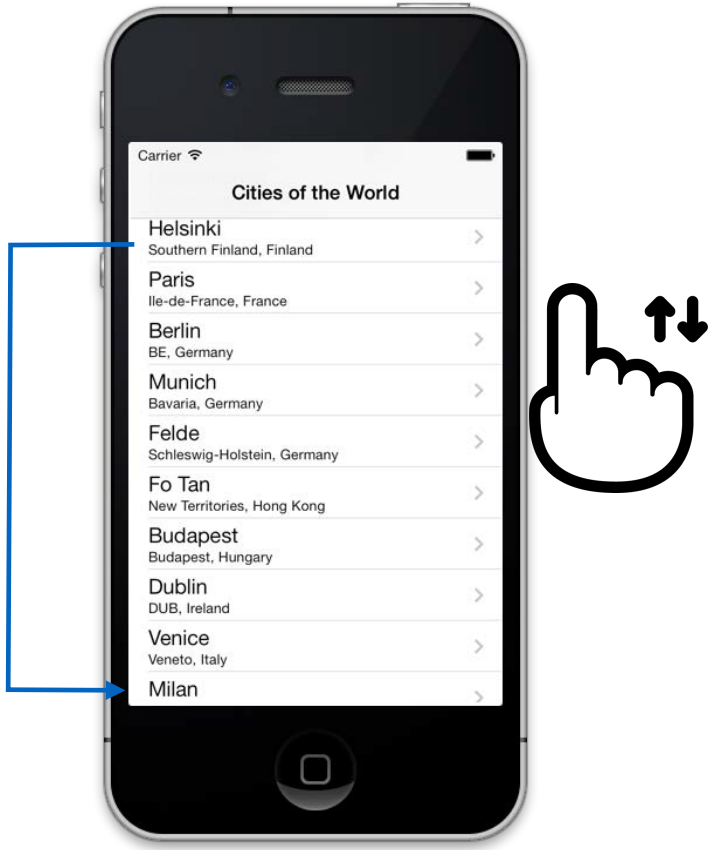


"I'll be your data source & delegate."



UITableView

The first cell is not longer visible so it can be reused for the next cell.



Common Patterns - Delegate

ViewController (delegate)

UITableView

```
Create new UITableView
Set ViewController as UITableViewDelegate
```

```
func tableView(numberOfRowsInSection)
return students.count
```

How many rows?

4

```
func tableView(cellForRowAtIndexPath)
return cell
```

What is in cell 1?

Flori 100000

```
func tableView(cellForRowAtIndexPath)
return cell
```

What is in cell 2?

Peter 100001

Source: Apple: Protocols

UITableViewDataSource

- three necessary methods to define the TableView
  - how many sections are in the table

```
func numberOfSectionsInTableView(tableView: UITableView) -> Int
```

  - how many rows are in each section

```
func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int
```

  - how should each cell be drawn

```
func tableView(tableView: UITableView, cellForRowAtIndexPath indexPath:
NSIndexPath)-> UITableViewCell
```

UITableViewDataSource

```
override func
numberOfSectionsInTableView(tableView: UITableView) -> Int {
    // Return the number of sections.
    return 1
}

override func
tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
    // Return the number of rows in the section.
    return students.count
}
```

UITableViewDataSource

```
func tableView(tableView: UITableView,
cellForRowAtIndexPath indexPath: NSIndexPath)-> UITableViewCell{

  //define the ReusableCell With an Identifier and as for
  //TypeSafty - Cell Identifier
  let cell =
            tableView.dequeueReusableCellWithIdentifier("StudentCell",
            forIndexPath: indexPath) as! StudentTableViewCell

  //get Object at row
  let student:Student = students[indexPath.row]
  // Configure the cell...
  cell.nameLbl.text = student.name

  return cell
}
```

UITableViewDataSource

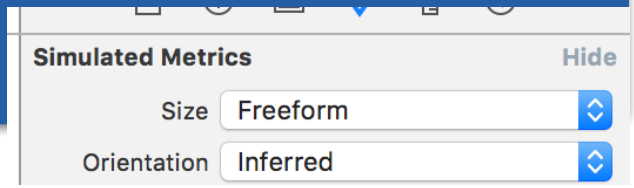
```
override func prepareForSegue(segue: UIStoryboardSegue, sender: AnyObject?) {
  //identif the seque
  if segue.identifier == “showStudentDetail"{
    //get selectedRow
    let indexPath:NSIndexPath = tableView.indexPathForSelectedRow()!

    //get selectedObject
    let student: Student = students[indexPath.row]

    //set viewController as next target
    let vc = segue.destinationViewController as DetailStudentViewController

    //set variable in viewController
    vc.student = student
    //in target viewDidLoad() handle further actions
  }
}
```

Datasource - Auto Height

```
override func
tableView(tableView: UITableView, estimatedHeightForRowAtIndexPath:IndexPath
section: Int) -> Int {
    return 100
}
```

- expensive! it is called for every cell. Maybe you can live with:

```
func viewDidLoad {
    …
    tableView.rowHeight = UITableViewAutomaticDimension
    tableView.estimateRowHeight = 100;
}
```



Delegate

- Controls how the table is displayed
- delegate observes what the table view is doing and how the user interacts with her

for example:

```
optional func tableView(_ tableView: UITableView,
willSelectRowAtIndexPath indexPath: NSIndexPath) -> NSIndexPath?
```

```
optional func tableView(_ tableView: UITableView,
didSelectRowAtIndexPath indexPath: NSIndexPath)
```

```
optional func tableView(_ tableView: UITableView,
willDeselectRowAtIndexPath indexPath: NSIndexPath) -> NSIndexPath?
```

```
optional func tableView(_ tableView: UITableView,
didDeselectRowAtIndexPath indexPath: NSIndexPath)
```

UITableView
Demo

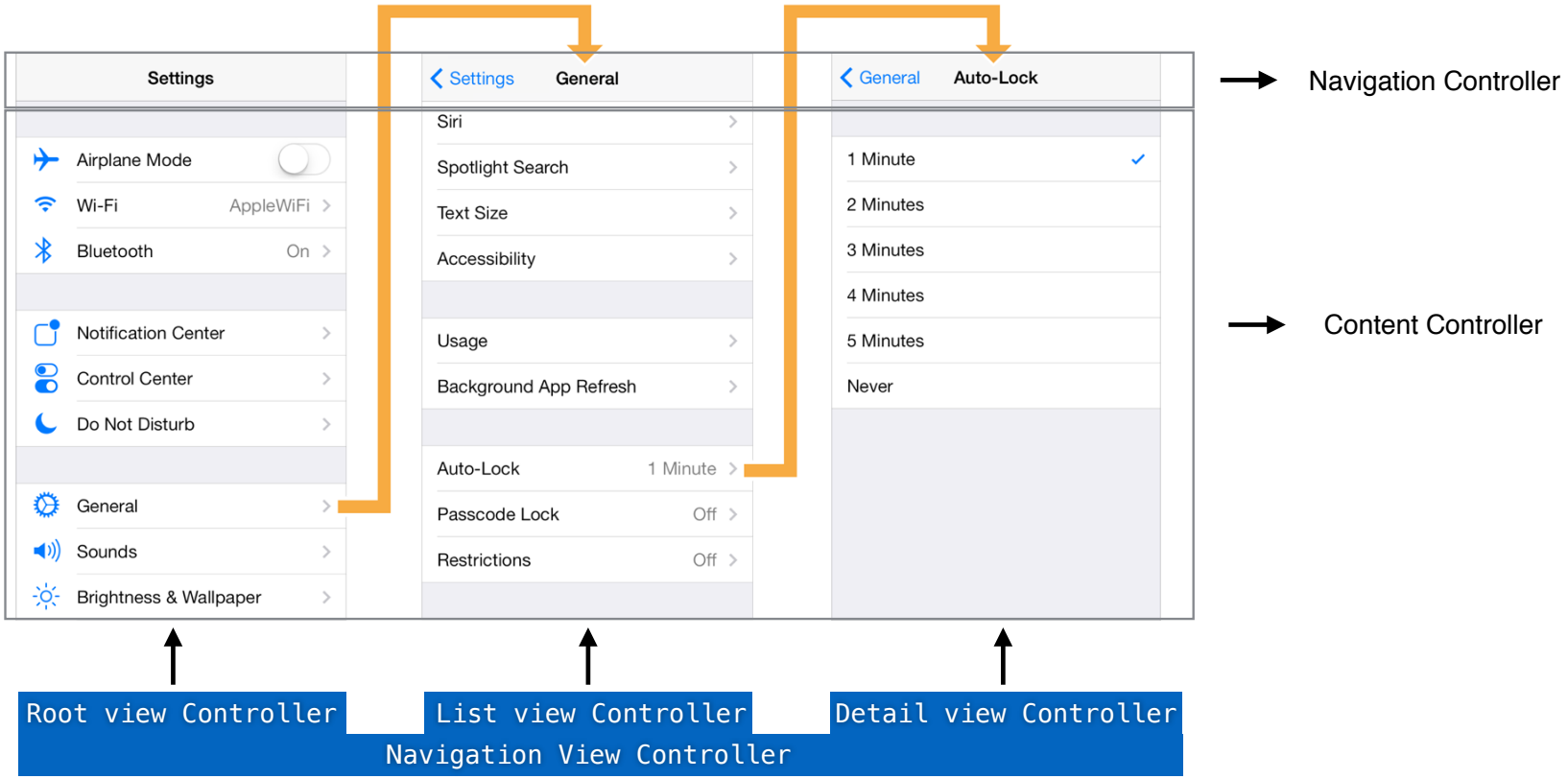
Table View

- How to navigate back to the table view from the detailViewController?
  - Option 1: Build an Segue
  - Option 2: Use `UINavigationController`

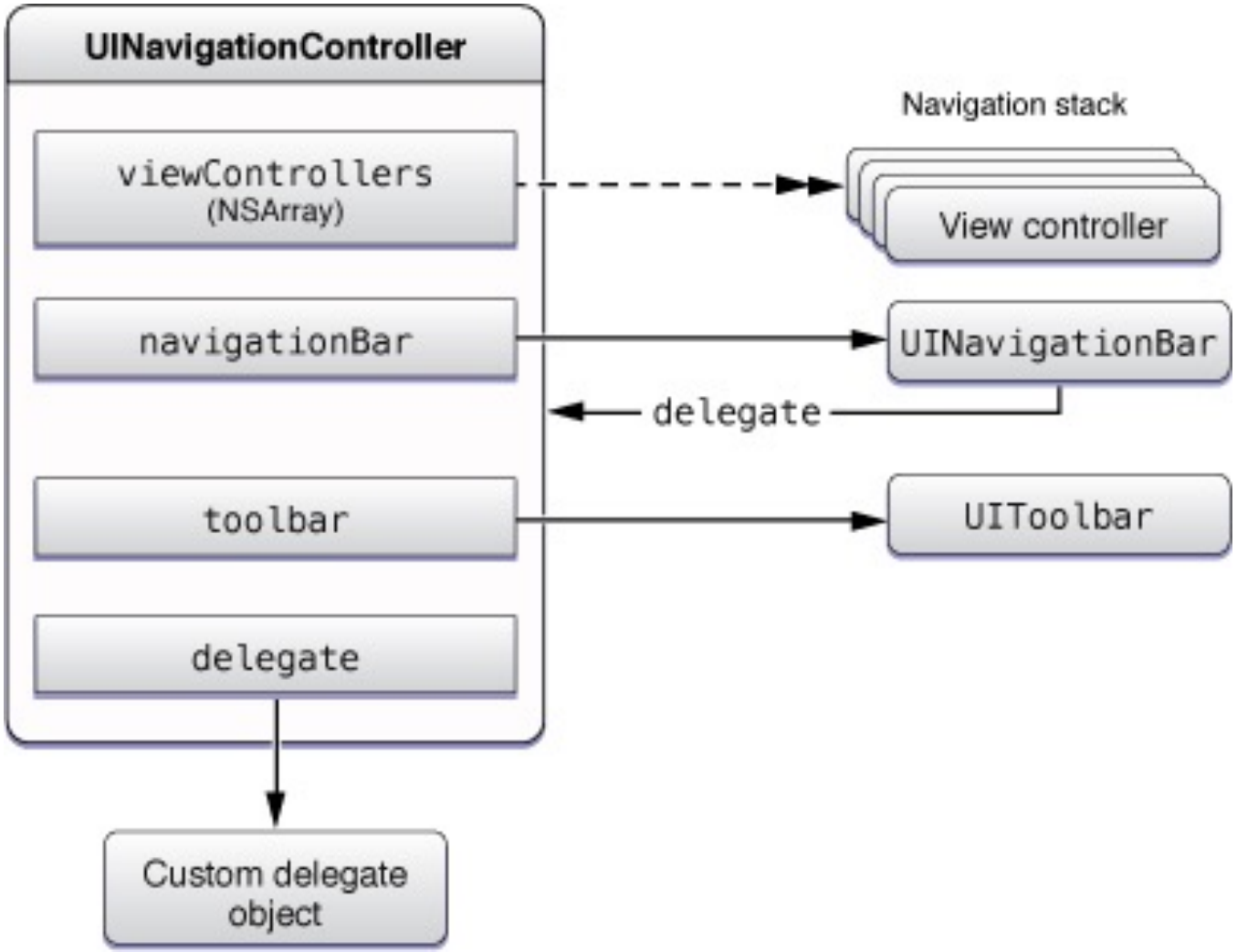
Navigation
UINavigationController

UINavigationController

- manages the navigation of hierarchical content



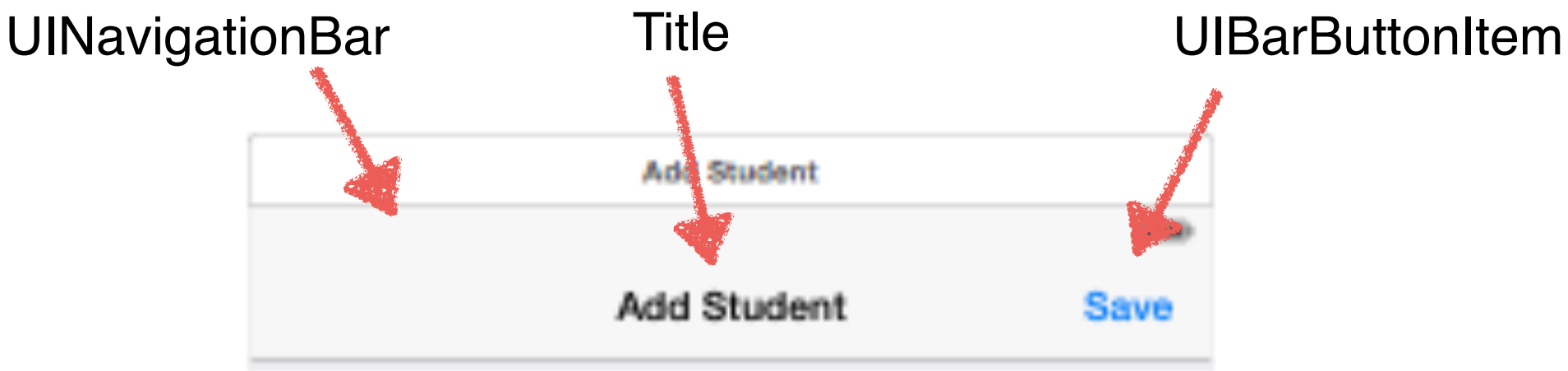
Source: Apple Developer Library

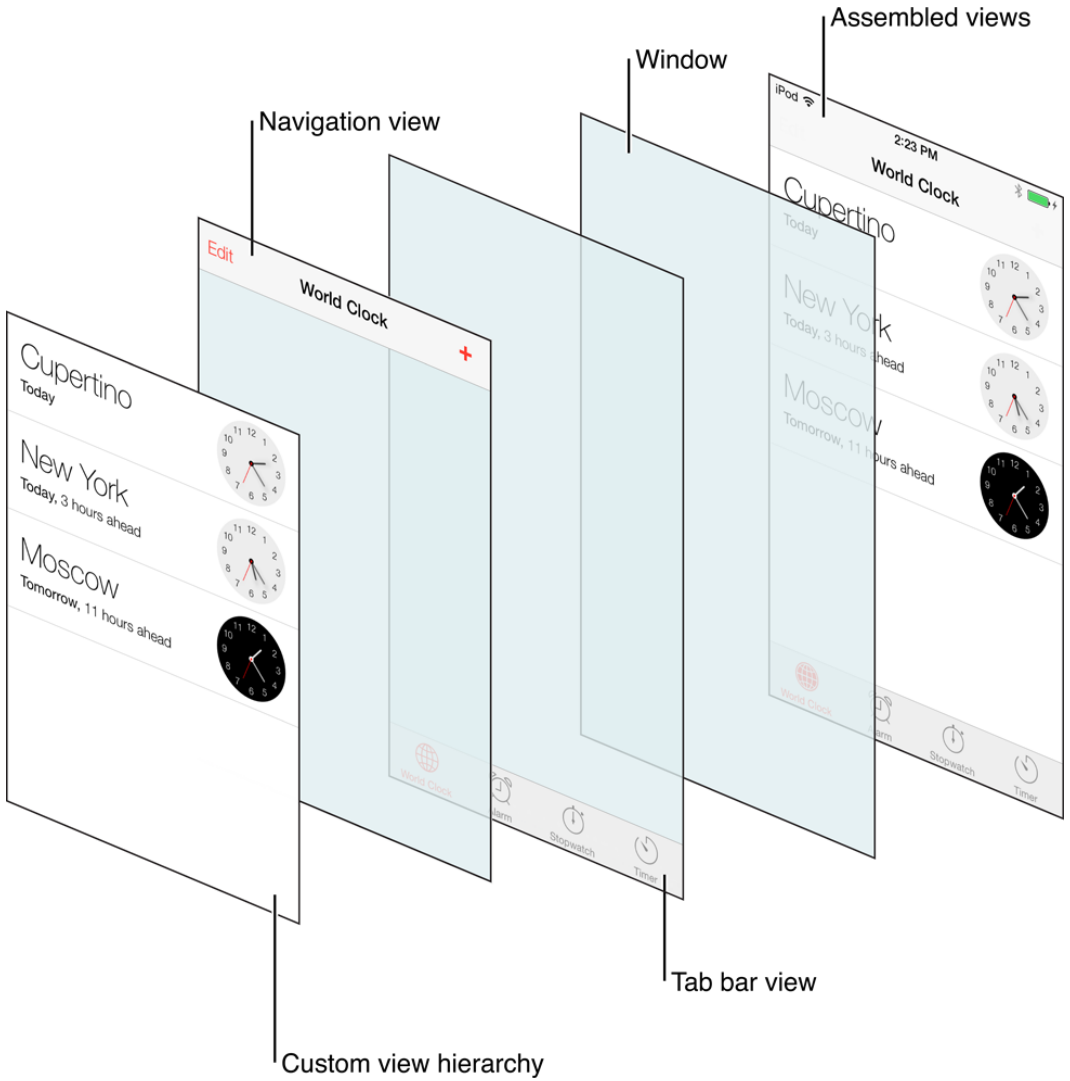


https://developer.apple.com/library/ios/documentation/UIKit/Reference/UINavigationController_Class/

UINavigationBar

- `UINavigationBar` offers a way to handle hierarchical user navigation





https://developer.apple.com/library/ios/documentation/UIKit/Reference/UINavigationController_Class/

UINavigationController

- Accessing Items on the Navigation Stack

```
topViewController
visibleViewController
viewControllers
setViewControllers(_:animated:)
```

- Pushing and Popping Stack Items

```
pushViewController(_:animated:)
popViewControllerAnimated(_:)
popToRootViewControllerAnimated(_:)
popToViewController(_:animated:)
interactivePopGestureRecognizer
```

https://developer.apple.com/library/ios/documentation/UIKit/Reference/UINavigationController_Class/

UINavigationController
Demo

UINavigationBar

- How to navigate back to the table view from the detailViewController?
  - Option 1: Build an Segue
  - Option 2: Use `UINavigationController`